CHAPTER 1

Introduction

Solutions for the Practice Exercises of Chapter 1

Practice Exercises

1.1

Answer:

Two disadvantages associated with database systems are listed below.

a. Setup of the database system requires more knowledge, money, skills, and time.

b. The complexity of the database may result in poor performance.

1.2

Answer:

a. Executing an action in the DDL results in the creation of an object in the database; in contrast, a programming language type declaration is simply an abstraction used in the program.

b. Database DDLs allow consistency constraints to be specified, which programming language type systems generally do not allow. These include domain constraints and referential integrity constraints.

c. Database DDLs support authorization, giving different access rights to different users. Programming language type systems do not provide such protection (at best, they protect attributes in a class from being accessed by methods in another class).

d. Programming language type systems are usually much richer than the SQL type system. Most databases support only basic types such as different types of numbers and strings, although some databases do support some complex types such as arrays and objects.

e. A database DDL is focused on specifying types of attributes of relations; in contrast, a programming language allows objects and collections of objects to be created.

1.3

Answer:

Six major steps in setting up a database for a particular enterprise are:

- Define the high-level requirements of the enterprise (this step generates a document known as the system requirements specification.)
- Define a model containing all appropriate types of data and data relationships.
- Define the integrity constraints on the data.
- Define the physical level.
- For each known problem to be solved on a regular basis (e.g., tasks to be carried out by clerks or web users), define a user interface to carry out the task, and write the necessary application programs to implement the user interface.
- Create/initialize the database.

1.4

Answer:

- **Data redundancy and inconsistency**. This would be relevant to metadata to some extent, although not to the actual video data, which are not updated. There are very few relationships here, and none of them can lead to redundancy.
- **Difficulty in accessing data**. If video data are only accessed through a few predefined interfaces, as is done in video sharing sites today, this will not be a problem. However, if an organization needs to find video data based on specific search conditions (beyond simple keyword queries), if metadata were stored in files it would be hard to find relevant data without writing application programs. Using a database would be important for the task of finding data.
- **Data isolation**. Since data are not usually updated, but instead newly created, data isolation is not a major issue. Even the task of keeping track of who has viewed what videos is (conceptually) append only, again making isolation not a major issue. However, if authorization is added, there may be some issues of concurrent updates to authorization information.

- **Integrity problems**. It seems unlikely there are significant integrity constraints in this application, except for primary keys. If the data are distributed, there may be issues in enforcing primary key constraints. Integrity problems are probably not a major issue.
- **Atomicity problems**. When a video is uploaded, metadata about the video and the video should be added atomically, otherwise there would be an inconsistency in the data. An underlying recovery mechanism would be required to ensure atomicity in the event of failures.
- **Concurrent-access anomalies**. Since data are not updated, concurrent access anomalies would be unlikely to occur.
- **Security problems**. These would be an issue if the system supported authorization.

1.5

Answer:

Queries used in the web are specified by providing a list of keywords with no specific syntax. The result is typically an ordered list of URLs, along with snippets of information about the content of the URLs. In contrast, database queries have a specific syntax allowing complex queries to be specified. And in the relational world the result of a query is always a table.